



Editor Introduction

Welcome to CodeWarrior.

This tutorial is designed to introduce the basic features and functionality of the Editor. The Editor is a major part of the CodeWarrior Integrated Development Environment (IDE).





Using the Controls

Use these three controls to move through this tutorial.

Previous

Click to move to the previous page.

Return to Topics

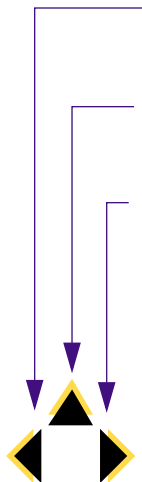
Click to return to the main topics page.

Next

Click to move to the next page.

Adobe Acrobat

The standard Adobe Acrobat commands are also available on the View menu.





Editor Overview

Before you get started, take a moment to review the information provided in this introduction before continuing. It may answer some of the questions you have on using this tutorial. The topics include:

- [Using the Controls](#)
- [Editor Topics](#)
- [Required Knowledge](#)
- [Intended Audience](#)
- [Taking the Tour](#)
- [Host Conventions](#)
- [Where Else to Go Tutorials](#)
- [Where Else to Go Documentation](#)





Editor Topics

This tutorial has two purposes. First, it introduces the CodeWarrior Editor features. Second, it guides you through common actions you need to know to effectively use the Editor. This tutorial includes:

- [Editor Window Tour](#)
- STAGE 1: [Editor Pop-ups](#)
- STAGE 2: [Editor's Other Components](#)
- [Editor Actions](#)
- STAGE 3: [Document Actions](#)
- STAGE 4: [Window Actions](#)
- STAGE 5: [Editing Actions](#)
- STAGE 6: [Navigating Actions](#)





Required Knowledge

This tour assumes that you have a basic understanding of using your computer system. If you are just getting started with computers, refer to the user guide that came with your computer system. Once you feel comfortable with using your computer, come back and start again.

NOTE: The CodeWarrior tutorials are designed to teach you how to use CodeWarrior to successfully create, compile, and debug executable code. These tutorials will not teach you how to program a computer. If you wish to learn programming, visit your local bookstore for beginning programming books, see your local community college for classes on programming, or refer to the 3rd-party documentation that may be provided on your CodeWarrior CD.





Intended Audience

The IDE Editor (Basic) tutorial is intended for all first-time code warriors. If you have not already taken the Guided Tour tutorial, we recommend that you take a few minutes to go through that tutorial before continuing here. The Guided Tour will give you an overview of the entire CodeWarrior IDE.

Beginning Programmers

If you are just beginning your career as a programmer then you are in for a treat. The CodeWarrior environment provides all the tools you need for any programming task, from simple applications to advanced professional-quality programs. Go through this tour, then follow the recommendations at [“Where Else to Go Tutorials”](#) to continue your explorations of CodeWarrior.

more...





Intended Audience (continued)

Experienced Programmers

After taking the Guided Tour tutorial, experienced programmers new to CodeWarrior should refer to this and the other tutorials for in-depth knowledge about the various components of the CodeWarrior environment. Many CodeWarrior Targeting manuals include basic tutorials that demonstrate how to use the tools to program for a particular platform.





Taking the Tour

The CodeWarrior tutorials are designed to help you become familiar with the CodeWarrior IDE. This tutorial consists of a series of stages. Each stage, in turn, is a series of slides, like this one. Each slide describes a feature or a task you can perform. You will gain the most benefit from these tutorials if you follow along with the CodeWarrior IDE while reading and performing the actions described in the tutorial.

To Use a Tutorial:

1. In Acrobat Reader 3.0, read the text on a slide.
2. If actions are described (example: Choose Open from the File menu), then switch to the CodeWarrior IDE.
3. In the CodeWarrior IDE, perform the action.
4. Switch back to Adobe Acrobat Reader, read the next action.

NOTE: Both Adobe Acrobat Reader and the CodeWarrior IDE must be running.





Host Conventions

Hosts: While the CodeWarrior IDE runs on several different host platforms, the functionality on these host platforms is not always identical. There are occasions when a feature is implemented for one or more hosts and not for others. This tutorial only demonstrates features found on all supported hosts.

Pictures: The IDE's visual interface on these hosts is nearly identical in all significant respects. When discussing a particular interface element such as a dialog box or window, the screenshot may come from any host. If you are using CodeWarrior on a different host, you should have no difficulty understanding the picture.

Keyboard: The default keyboard shortcuts for CodeWarrior on these platforms are very similar. However, keyboards and shortcuts do vary across host platforms. For example, a typical keyboard for a Windows machine has an Alt key, but a typical keyboard for a Mac OS computer calls the same key the Option key

more...





Host Conventions (continued)

To handle these kinds of situations, CodeWarrior documentation identifies and uses the following paired terms in the text:

- **Enter/Return** - the “carriage return” or “end of line” key. This is not the numeric keypad Enter key, although in almost all cases that works the same.
- **Backspace/Delete** - the key that (in text editing) causes the character before the insertion point to be erased (not to be confused with **Delete/Del**, the “forward delete” key)
- **Ctrl/Command** - in most cases, CodeWarrior maps the Windows Ctrl key the same way as the Mac OS Command key.
- **Alt/Option** - in most cases, CodeWarrior maps the Windows Alt key the same way as the Mac OS Option key.

For example, you may encounter instructions such as “Press **Ctrl/Command+S** to save a file,” or “**Alt/Option** click the Function pop-up to see the functions in alphabetical order.” Use the appropriate key as it is labeled on your keyboard.





Where Else to Go Tutorials

Once you have finished this tutorial, be sure to explore the other CodeWarrior tutorials, and the documentation that accompanies this product. For an overview of the IDE, see:

- [Guided Tour](#)—overview of the CodeWarrior IDE
- [Project Manager \(Basic\)](#)—introduces the Project Manager
- [Editor \(Basic\)](#)—introduces the CodeWarrior Editor
- [MW Debugger \(Basic\)](#)—introduces debugging features
- [Find & Replace](#)—introduces Find and Replace features





Where Else to Go Documentation

To learn more about programming for a particular target operating system or processor, see the corresponding CodeWarrior Targeting manual. The CodeWarrior IDE Editor features are also documented in:

- IDE User Guide—documents IDE components, including the CodeWarrior Editor.
- Targeting manual for your programming target. Examples include Targeting Windows, Targeting Rhapsody, and Targeting Palm OS.





Getting Started

Phew! Now that all of the preliminaries are out of the way, its time to explore the CodeWarrior Editor.





Editor Window Tour

This section identifies the major features available in the Editor and describes their use. Along the way you will discover:

- STAGE 1: [Editor Pop-ups](#)
- STAGE 2: [Editor's Other Components](#)



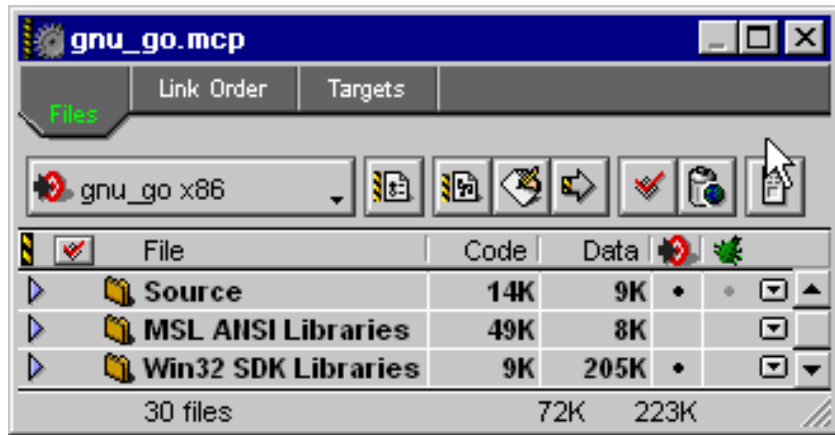


Getting Ready

You will use the `gnu_go.mcp` project file throughout this tutorial. If you don't already have it open, please open the project now.

The `gnu_go.mcp` project file is located in the Source Exam-

amples folder inside the Core Tutorials folder.





Editor Pop-ups

This section identifies the pop-up components of an Editor window and describes their use.

Along the way you will encounter:

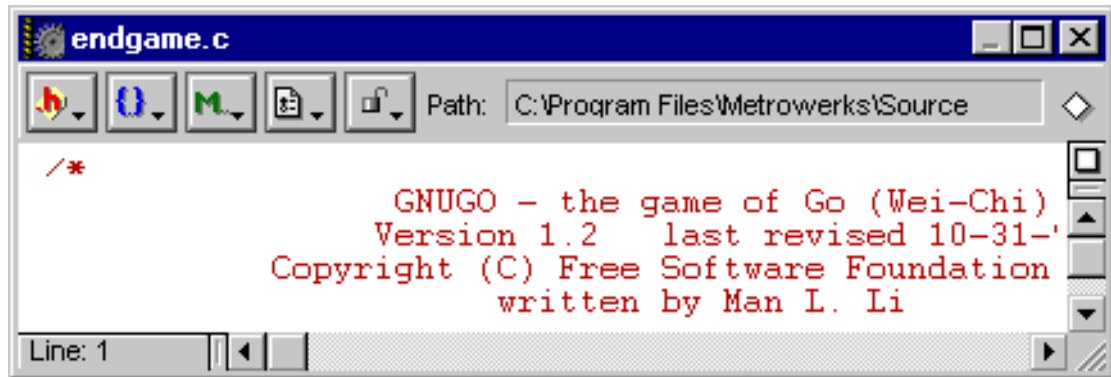
- [Interfaces Pop-up](#)
- [Routines Pop-up](#)
- [Marker Pop-up](#)
- [Options Pop-up](#)
- [VCS Pop-up](#)





Getting Ready

You will use the `endgame.c` file located in the `gnu_go.mcp` project file throughout this tutorial. If this file is not open, double-click the `endgame.c` file in the Project window. An editor window opens, and `endgame.c` becomes the current or active file.





Interfaces Pop-up

The Interfaces pop-up menu lists every interface (header) file referenced by the current file. You use this pop-up to open an interface file. In this step you open the `gnugo.h` file using the Interface pop-up in the `endgame.c` editor window.

To Open an Interface or Header File:

- Choose `gnugo.h` from the Interface pop-up.

An Editor window appears containing the `gnugo.h` interface file.

Close the `gnugo.h` editor window before continuing to the next slide.





Routines Pop-up

The Routines pop-up menu lists every routine referenced in the current file. You can use this pop-up to scroll to the chosen routine, and the Editor positions the text insertion point in that routine. In this step you reposition the text insertion point in the `endgame.c` file.

To Move to a Different Routine:

- Choose `endgame` from the Routine pop-up.

The `endgame ( )` routine appears in the text editing area.

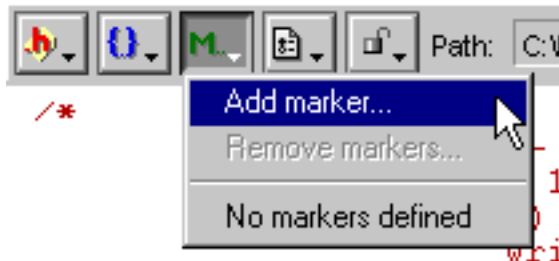
Also, note the position of the text insertion point.





Marker Pop-up

You use the Marker pop-up to create custom markers in a file for quick and easy navigation. A marker is a bookmark that identifies a location in a source file. You assign a name to the marker to make navigating to it easier.



For information on adding, using and removing markers, see “Navigating with Markers.”





Options Pop-up

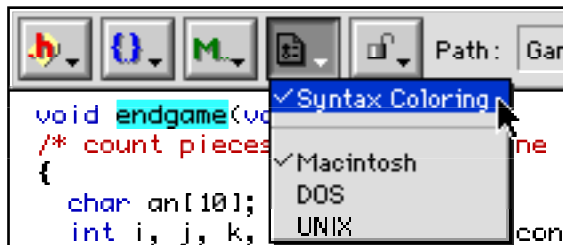
You use the Options pop-up menu to choose syntax highlighting for the current file, and to set the end-of-line format when saving the file.

Syntax Coloring—enables or disables the syntax highlighting of text within the Editor window.

To Enable or Disable Syntax Coloring:

- Choose **Syntax Coloring** from the **Options** pop-up.

When you choose Syntax Coloring for endgame.c, the coloring state changes. Choose it again to restore the window to its previous state.





File Format Options

You can also specify the end-of-line (EOL) format CodeWarrior should use when saving the file. The checkmark indicates the file format currently used by the file.

Macintosh—stores files using the Macintosh format (CR).

DOS—stores files using the DOS format (CR/LF).

UNIX—stores files using the UNIX format (LF).

To Set a File's EOL Format:

- Choose **Macintosh**, **DOS**, or **UNIX** from the **Options** pop-up.

The Editor will use the chosen EOL file format the next time you save the file.





VCS Pop-up

You use the VCS pop-up to manage files stored in a version control system. This pop-up does nothing if a version control system is not installed and enabled.



For more information on using the VCS pop-up see the IDE User Guide and the documentation that came with the VCS system.





Summary

This concludes the Editor pop-up stage of this tutorial.

In this stage you learned how to:

- Use the Interfaces pop-up
- Use the Routines pop-up
- Use the Options pop-up
- Identify the Marker and VCS pop-ups

The ability too quickly open any included file, jump to any routine, or navigate to any marked location in a file lets you spend more time writing code, and less time finding it.

Now that you understand the pop-ups present in all Editor windows, its time to investigate the other features found in Editor windows.





Editor's Other Components

This section of the tutorial identifies and describes the other components that appear in an Editor windows.

Along the way you will identify and use the:

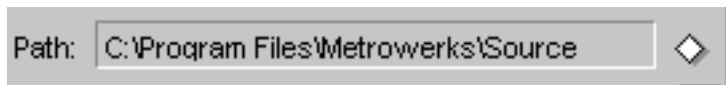
- [File Path Caption](#)
- [File Dirty Marker](#)
- [Toolbar Control](#)
- [Pane Splitter Controls](#)
- [Line Number Control](#)





File Path Caption

The File Path caption box displays the full pathname for the file displayed in the Editor window. This makes it easy to identify the location of a particular file.





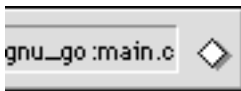
File Dirty Marker

The File Dirty marker indicates whether the file in the Editor window requires saving or not. In this step you modify the `main.c` file to change its File Dirty status.

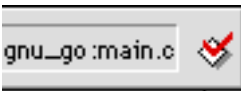
To Modify a File:

- Enter “Test” in the first line of the `main.c` file.

The marker changes to indicate the document needs saving.



File has not been modified.



File modified and requires saving.





Toolbar Control

Use the Toolbar control to alternate the position of the Editor window toolbar. When the alternate toolbar position is in effect, the Interface, Routines, Markers, Options, and VCS pop-up menus appear in the bottom left corner of the window.



Toolbar control
switches the
position of the
Editor toolbar

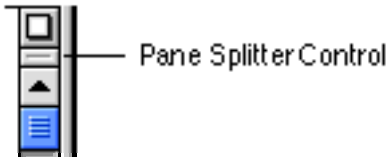
For more information on switching toolbar views, see [Switching Toolbars](#).





Pane Splitter Controls

You use the Pane Splitter controls to view two or more areas of the same document within the same Editor window. Panes can be created, resized, and removed as necessary. Panes can be vertical or horizontal in nature depending upon the Pane Splitter control utilized.



For more information on using Pane Splitter controls, see [Managing Multiple Panes](#).





Line Number Control

The Line Number control displays the line number the cursor is currently positioned in and enables you to position the text insertion point in another line of the same file. In this step you reposition the text insertion point using the Line Number control.

To Jump to a Specific Line:

1. Click the **Line Number** control.

The Line Number dialog box appears.

2. Enter a line number in the **Line Number** dialog box. Try “50.”
3. Click **OK**.

The Editor window displays the specified line and positions the text insertion point at the start of the line.





Summary

This concludes the Other Editor Components stage of this tutorial.

In this stage you learned how to:

- Identify the location of the file
- Identify a file that needs saving
- Hide, show, and save the toolbar state
- Add, resize, and remove an editable text pane
- View and change the current line number

Now that you can identify and use all of the Editor components, its time to explore the actions you can perform within an Editor window.





Editor Actions

This section of the tutorial identifies the components of an Editor window and describes their use.

Along the way we will cover:

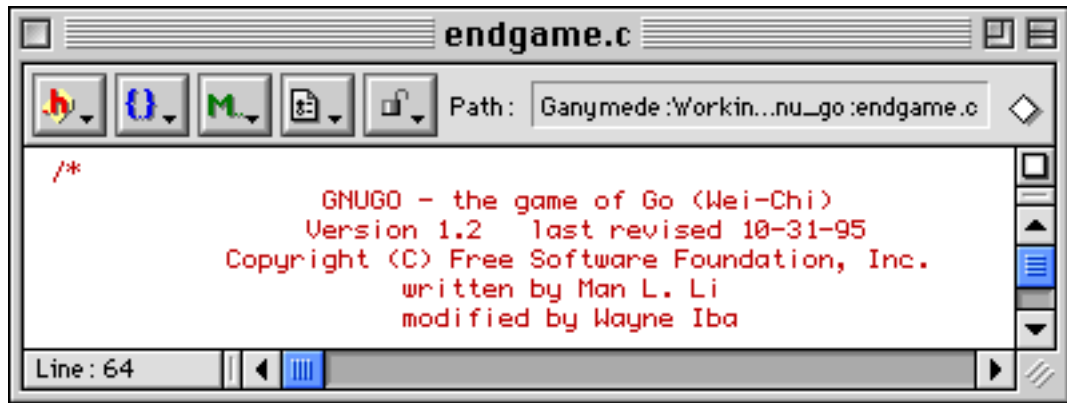
- STAGE 3: [Document Actions](#)
- STAGE 4: [Window Actions](#)
- STAGE 5: [Editing Actions](#)
- STAGE 6: [Navigating Actions](#)





Getting Ready

You will use the `endgame.c` file located in the `gnu_go.mcp` project file throughout this tutorial. If this file is not open, double-click the `endgame.c` file in the Project window. An editor window opens, and `endgame.c` becomes the current or active file.





Document Actions

This section describes the various document actions used by the Editor.

Along the way we will cover:

- [Creating Documents](#)
- [Saving Documents](#)
- [Reverting Documents](#)





Creating Documents

You create a new document when you want to start a new source code file.

To Create a New Document:

- Choose **New** from the **File** menu.

The IDE creates a new, empty Editor window.





Saving Documents

You save a document to retain changes between editing sessions. The IDE has three ways to save a document. Each has a slightly different effect.

You save a document to save changes made during an editing session.

To Save a File Using the Same Name:

- Choose **Save** from the **File** menu, **OR**
Type the keyboard shortcut Ctrl/Command+S.

The changed file is saved to disk.

To Save a File With a New Name:

1. Choose **Save As** from the **File** menu.

The standard save dialog box appears.





Saving Documents (continued)

2. Enter a new name in the save dialog box, click **Save**.

The file is saved to disk, and the Editor and Project windows update to reflect the new name.

You can save a copy of a file to provide a backup of the original file before you begin making changes.

To Save Copy of a File:

1. Choose **Save A Copy As** from the **File** menu.

The standard save dialog box appears.

2. Enter a new name in the save dialog box.

A copy of the file is saved to disk.





Reverting Documents

You revert a document to replace changes made to the open file with the contents of the last version saved to disk.

To Revert a Document:

- Choose **Revert** from the **File** menu.

The file on disk replaces the current file in the Editor window.





Summary

This concludes the Document Actions stage of this tutorial.

In this stage you learned how to:

- Create a new document
- Save documents
- Revert documents

Now that you have mastered the document actions, in the next stage you'll work with windows, editing, and navigation. Let's look at some of the things you can do with an Editor window.





Window Actions

This stage of the tutorial identifies the Editor window actions.

Along the way we will cover:

- [Switching Toolbars](#)
- [Saving Window Settings](#)
- [Managing Multiple Panes](#)





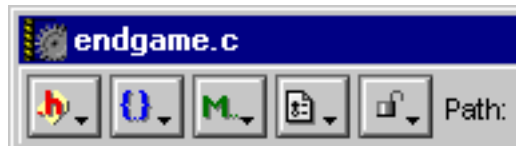
Switching Toolbars

You switch the position of the Toolbar in an Editor window by using the [Toolbar Control](#). Which configuration you use is a matter of personal preference, it doesn't matter to the CodeWarrior IDE.

To Relocate The Editor Toolbar:

- Click the Toolbar control.

The toolbar changes its position in the Editor window.



Toolbar at top of Editor window



Toolbar at bottom of Editor window





Saving Window Settings

You can save the Editor window's settings, which includes the position of the toolbar. When you do, all Editor windows thereafter use the same settings.

To Save the Window Settings:

1. Make the desired window active.
Click the window you want to use as a model for future Editor windows.
2. Choose **Save Default Window** from the **Window** menu.
The IDE saves the Editor window configuration.



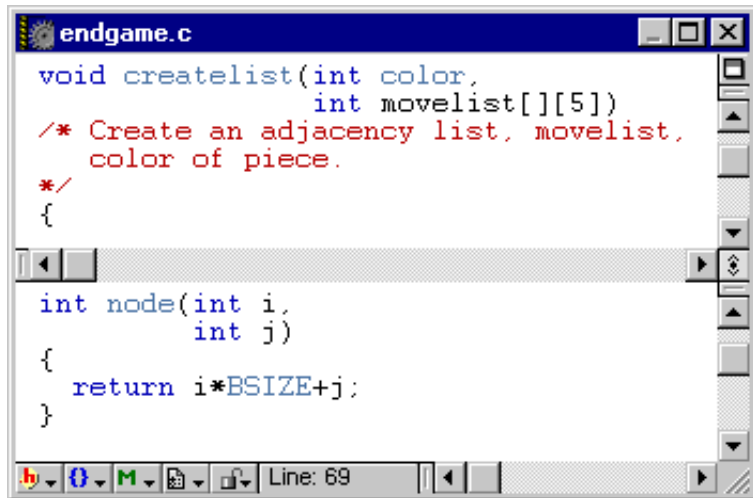


Managing Multiple Panes

You can split an Editor window into multiple panes. Panes allow you to view two or more portions of the same document an Editor window. Panes can be created, resized, and removed as necessary.

The pane actions include:

- [Adding a Pane](#)
- [Adjusting Panes](#)
- [Removing Panes](#)





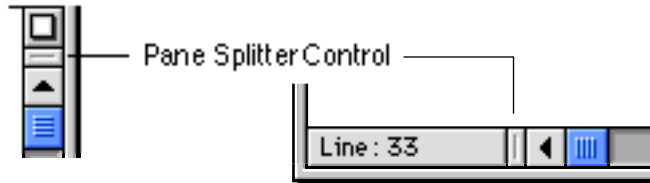
Adding a Pane

You use a Pane Splitter control to create a new pane. In this step you create a new pane in the endgame.c Editor window.

To Add a Pane:

- Double-click the vertical pane splitter control, **OR**
Drag the vertical pane splitter control.

A second editable text pane appears in the Editor window.



NOTE: You can also split a window or pane horizontally.





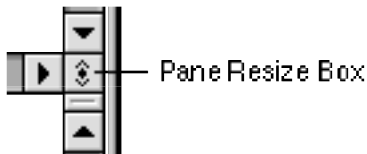
Adjusting Panes

You adjust the size of a pane as necessary to view the amount of text desired. In this step you adjust a pane's size in the `endgame.c` Editor window.

To Adjust a Pane:

- Drag the vertical pane resize box to adjust the pane size.

The pane's are resized in the Editor window.





Removing Panes

You can remove panes from an Editor window using the Pane Resize box. In this step you remove a pane from the `endgame.c` Editor window.

To Remove a Pane:

- Double-click the vertical pane resize box, **OR**
Drag the vertical pane resize box to the edge of the Editor window.
The second editable text pane disappears in the Editor window.





Summary

This concludes the Window Actions stage of this tutorial.

In this stage you learned how to:

- Reposition the Editor window's toolbar
- Save the Editor window configuration
- Create, resize, and remove panes

With Window Actions under your belt, its time to look into all the Editing Actions you can perform in an Editor window.





Editing Actions

This stage of the tutorial describes many common Editing Actions that you can perform in an Editor window.

Along the way we will cover:

- [Adding Text](#)
- [Selecting Text](#)
- [Deleting Text](#)
- [Dragging Text](#)
- [Copying Text](#)
- [Shifting Text Left and Right](#)





Adding Text

Adding text to a CodeWarrior file is just as easy as adding text to any word processor or text editor. In this step you add text to the `endgame.c` source file.

To Add Text:

1. Place the insertion cursor.

Try the line above the one that reads: `#include <stdio.h>`. The insertion cursor appears at the text entry point.

2. Enter text.

Type `/* insertion test */`. The text appears in the Editor window.





Selecting Text

You select text to cut, copy, move, or delete it from the document. In this step you select text in the `endgame.c` source file.

To Select Any Text:

- Drag the mouse, releasing the mouse button when finished.

Select `/* insertion test */` by dragging the mouse over it.

To Select a Single Word

- Double-click the word.

Select the word “insertion” is by double-clicking it.

To Select a Line of Text

- Triple-click a line.

Select the line `/* insertion test */` by triple-clicking it.





Deleting Text

In this step you delete text in the endgame.c source file.

To Delete Text:

1. Select the text to be deleted.
Select the text `/* insertion test */` in the document.
2. Choose **Cut** or **Clear** from the **Edit** menu, **OR**
Press the Delete key.
The selected text disappears.





Dragging Text

You can use traditional cut and paste techniques to move text in the same file or to a different file. You can also simply drag selected text to accomplish the same goals. In this step you drag text to move it in the `endgame.c` source file.

To Drag Text:

1. Select the text.

Select the line: `#include <stdio.h>`.

2. Drag the text to a new position.

A focus line indicates the insertion point of the text.





Copying Text

Using dragging, you can copy text to a new position in the same, or another document file. In this step you copy a line of text in the `endgame.c` source file.

To Copy Text:

1. Select the text.

Select the line: `#include <stdio.h>`.

2. On Windows, press and hold the Control key down, **OR** On the Mac OS, press and hold the Option key down.
3. Drag the text copy to the new position.

A focus line indicates the insertion point of the text.





Shifting Text Left and Right

You shift text right or left to control the indentation of lines of code. The amount of indentation is specified by the Tab Size field in the Font & Tab preference panel. In this step you shift some lines of text in the `endgame.c` source file.

To Shift Text Right:

1. Select the text.

Select the lines `#include <stdio.h>` through `#include "gnugo.h"`.

2. Choose **Shift Right** from the **Edit** menu, **OR**
Type the keyboard shortcut `Ctrl/Command+] (right bracket)`.

The selected text lines are shifted right in the document.





Shifting Text Left and Right (continued)

To Shift Text Left:

1. Select the text.

Select the lines `#include <stdio.h>` through `#include "gnugo.h"`.

2. Choose **Shift Right** from the **Edit** menu, **OR**
Type the keyboard shortcut `Ctrl/Command+[` (left bracket).

The selected text lines are shifted left in the document.





Summary

This concludes the Editing Actions stage of this tutorial.

In this stage you learned how to:

- Add, select, and delete text in an Editor window
- Move and copy selected text by dragging
- Shift text left and right

Now that you've finished the Editing Actions stage, you're ready for the final stage of this tutorial where you learn how to navigate quickly and easily through your code.





Navigating Actions

Moving around among files and within a file is critical to enhancing your productivity as a programmer. The CodeWarrior IDE has several unique features. You have already met a few, such as the Interface and Routine pop-ups. In this stage you learn ways to move around, and how to manage the insertion point.

Along the way we will cover:

- [Navigating Files](#)
- [Navigating Source Code](#)
- [Navigating with Markers](#)





Navigating Files

In the Project Manager tutorial you saw how to use group pop-ups in the Project window to open files contained in the group. You also used the file Interface pop-ups to open header files associated with a specific file.

Well, there is another method to open files, its called:

- [Opening Files with Open Selection](#)





Opening Files with Open Selection

You can open any file name appears in an Editor window using Open Selection. In this step you open the `stdio.h` file.

Open File with Open Selection:

1. Select `stdio.h` in the `endgame.c` file.

When selecting a file name, make sure not to include any brackets or angle characters that enclose the name.

```
#include <stdio.h>
#include <string.h>
#include "gnugo.h"
```

Choose **Open Selection** from the **File** menu, **OR**
Type the keyboard shortcut **Ctrl/Command+D**.

CodeWarrior searches for the file, displaying it in a separate Editor window when found.





Navigating Source Code

You can use the keyboard to navigate your source code. The following is a table of common navigational actions and the keyboard commands that invoke them. Take a moment to investigate and try out these actions on the `main.c` file.

Action	Keyboard Command
Top of document	Ctrl+Home/Command+Up Arrow
Bottom of document	Ctrl+End/Command+Down Arrow
Top of page	Page Up/Option+Up Arrow
Bottom of page	Page Down/Option-Down Arrow
Start of line	Home/Command+Left Arrow
End of line	End/Command-Right Arrow
Previous word	Ctrl/Option-Left Arrow





Action	Keyboard Command
Next word	Ctrl/Option+Right Arrow
Scroll line up (does not move cursor)	Ctrl/Control+Up Arrow
Scroll line down (does not move cursor)	Ctrl/Control+Down Arrow
Move cursor up a line	Up Arrow
Move cursor down a line	Down Arrow

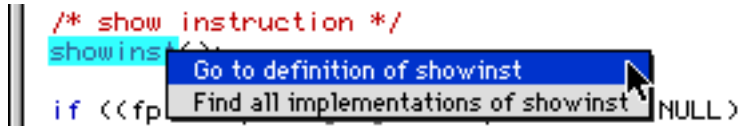
NOTE: See the IDE User Guide for more information on other keyboard commands.



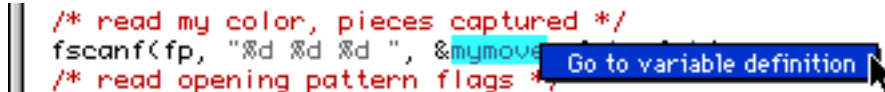


Context Pop-ups

You use context pop-ups to display variable and routine definitions, as well as to view all implementations of a routine's usage. In this step you use a context pop-up to view a routine definition.



Context pop-up for routines



Context pop-up for variables





Context Pop-ups (continued)

To View a Context Pop-up:

1. Click on the routine `showboard` in `endgame.c`.
The context pop-up appears after a short delay.
2. Choose **“Go to definition of showboard”** from the pop-up.
The `showboard.c` file appears in an Editor window with the `showboard` routine highlighted.

NOTE: Activate Browser must be enabled to use contextual menus in the Editor.





Navigating with Markers

Markers provide customizable bookmarks that make navigating source code quick and easy.

The Marker actions include:

- [Adding Markers](#)
- [Using Markers](#)
- [Removing Markers](#)





Adding Markers

You add a marker to identify a location in a file. This location can be anything in the file, a routine, a line of code, a remark, even a blank line. In this step you add a new marker to the `endgame.c` file.

To Add a Marker:

1. Place the cursor on the line the marker will identify.

Locate the blank line directly before the `/* count total */;` remark near the end of the `endgame.c` file.

2. Choose **Add Marker** from the **Marker** pop-up.
3. Enter a marker name in the **Add Marker** dialog box. Try “Test Marker”.
4. Click **Add**.

The name **Test Marker** appears in the Marker pop-up.





Using Markers

You navigate to a marker by choosing it from the Marker pop-up menu. In this step you use the marker created in [Adding Markers](#).

To Use a Marker:

- Choose the marker name from the **Marker** pop-up.

The marker is displayed in the Editor window.





Removing Markers

When a marker becomes obsolete, you can remove it from the Marker pop-up using the Remove Marker dialog box. In this step you delete the marker created in [Adding Markers](#).

To Remove a Marker:

1. Choose **Remove Marker** from the **Marker** pop-up.
2. Choose “Test Marker” in the **Remove Markers** dialog box.
3. Click **Remove**.

To remove more than one marker, repeat Steps 2 and 3.

4. Click **Done**.

The Remove Marker dialog closes.





Summary

This concludes the Navigation Actions stage of this tutorial.

In this stage you learned how to:

- Open files from the Editor window
- Navigate source files using keyboard shortcuts
- Add, use, and remove markers

With Window Actions under your belt, its time to look into all the Editing Actions you can perform in an Editor window.





Conclusion

Congratulations!

You have completed the CodeWarrior Editor tutorial. You have met the interface elements of the Editor, and learned how to use them to open files, locate and jump to both predefined and custom markers. While there is still more to learn, you've taken another step on your path to being a successful code warrior.

Now that you understand the Editor, see the ["Where Else to Go Tutorials"](#) slide for information on other tutorials and documentation available.





Technical Support

If you should encounter any difficulties with this guided tour, contact Metroworks Technical Support for assistance. Please use the appropriate report form located in the Release Notes folder on the CD. They can be reached at:

`support@metrowerks.com`





Copyright Information

Metrowerks CodeWarrior copyright ©1993–1998 by Metrowerks Inc. and its licensors. All rights reserved.

Documentation stored on the compact disk(s) may be printed by licensee for personal use. Except for the foregoing, no part of this documentation may be reproduced or transmitted in any form by any means, electronic or mechanical, including photocopying, recording, or any information storage and retrieval system, without permission in writing from Metrowerks Inc.

Metrowerks, the Metrowerks logo, CodeWarrior, and Software at Work are registered trademarks of Metrowerks Inc. PowerPlant and PowerPlant Constructor are trademarks of Metrowerks Inc.

All other trademarks and registered trademarks are the property of their respective owners.

ALL SOFTWARE AND DOCUMENTATION ON THE COMPACT DISK(S) ARE SUBJECT TO THE LICENSE AGREEMENT ON THE CD.

Revision Code: 01/14/98-lft

